

**ATTORNEY DOCKET NO.**  
**13644.0046**

**PATENT**

**REMARKS/ARGUMENTS**

Claims 1-3 and 5-8 stand rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Patent 5,752,038 to Blake et al. Claim 4 stands rejected under 35 U.S.C. 103(a) as being unpatentable over Blake in view of U.S. Patent 6,115,809 to Mattson et al. Claims 9 and 10 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Blake in view of U.S. Patent 6,634,023 to Komatsu et al. These rejections are respectfully traversed.

**Rejections under 35 U.S.C. 102**

Claims 1-3 and 5-8 stand rejected under 35 U.S.C. 102(b) as being anticipated by Blake. In particular, it is alleged that Blake discloses placing a first plurality of directives in the source code to divide the computer program into the program objects, whereby an annotated computer program is produced, and processing the annotated computer program to generate a description for each of the objects prior to generating an executable image of the annotated computer program. This rejection is respectfully traversed.

Blake fails to provide a basis for the rejection of claims 1-3 and 5-8 under 35 U.S.C. 102(b), because it fails to disclose each element of the claimed invention. For example, the Examiner alleges that Blake discloses placing a first plurality of directives in the source code to divide the computer program into the program objects, whereby an annotated computer program is produced at col. 5, lines 18-20. However, it is noted that the cited section of Blake states that "the compiler program 110 automatically inserts a call to the library routine into each code portion while compiling the source module 116." As such, Blake discloses a process that is occurring during the step of claim 1 of "generating an executable image of the annotated computer program, wherein the executable image is configured for execution on the target computer platform," whereas it is clear from claim 1 that the step of "placing a first plurality of directives in the source code to divide the computer program into the program objects" occurs before compilation, as the annotated computer program must first be ported to the target computer platform before the executable image is generated. Consistent terminology is used in Blake, note step 202 of Fig. 2 of Blake, which recites "compile and link source modules to produce instrumented executable module." As such, Blake fails to disclose the claimed element.

**ATTORNEY DOCKET NO.**  
**13644.0046**

**PATENT**

Furthermore, Blake only discloses inserting a call to a library routine while the program is being compiled into an executable, whereas the examples of a "directive" disclosed in the specification of the present application at page 10 include multiple functions that do not include a call to a library routine. While a call to a library routine could be one of the plurality of directives, it is clear that a directive must be more than a call to a library routine. Construing a directive to be nothing more than a call to a library routine is improper, as it fails to "indicate linkages between the program objects," as disclosed at page 9, lines 21-22 of the specification.

It is further alleged that Blake discloses processing the annotated computer program to generate a description for each of the program objects at col. 5, lines 43-44. However, it is again noted that the cited section of Blake that allegedly discloses this step states that this occurs "during execution of the instrumented executable module." In contrast, the claimed invention performs processing of the annotated computer program to generate a description for each of the program objects before program objects are allocated to fixed locations in the memory of the target computer platform, before the annotated computer program is ported to the target computer platform, and before the executable image of the annotated computer program, which is configured for execution on the target computer, is generated. As such, Blake fails to disclose the claimed element.

It is also alleged that Blake discloses allocating the program objects to fixed locations in the memory of the target computer platform at Fig. 6, item 608 and the corresponding sections of the disclosure. However, it is clear from Blake that Fig. 6 discloses a process that is performed on a bit vector table of a compiled execution data file, as opposed to an uncompiled annotated computer program. As such, Blake fails to disclose the claimed element.

Claim 2 includes "the program objects comprise executable code, constant data, and volatile data." The Examiner construes this claim to be what is disclosed by Blake at col. 4, lines 1-3. However, not only are the words "constant" and "volatile" not even used anywhere at all in Blake, much less in the cited section, the assertion by the Examiner is contrary to the plain meaning of the claim. The Examiner asserts that the "executable code would include constant and volatile data," but the claim wording states that "the program objects comprise executable code, constant data, and volatile data," which means that the executable code, constant data and volatile data are separate program objects, not all contained within a single object as asserted

**ATTORNEY DOCKET NO.**  
**13644.0046**

**PATENT**

by the Examiner. The Examiner's construction would be correct if the claim read "one or more program object comprises executable code, constant data, and volatile data," or "each program object comprises executable code, constant data, and volatile data," but claim 2 simply does not read like that. This construction is confirmed by reference to the specification at page 8, lines 1-3, which states that each "of these three entities, executable code, constant data, and volatile data, constitutes a class of objects." Thus, the Examiner's construction is not only contrary to the plain meaning of the claim language, it is also contrary to the exemplary embodiments in the specification.

Claim 3 includes the method of claim 1 "further comprising the step of estimating a typical usage for each of the program objects." Again, the Examiner misconstrues this to be what is disclosed at col. 4, line 7 of Blake, which pertains to the point in time relative to other program modules at which a code portion is used; however, that is not the ordinary meaning of the term "typical usage." Referring to page 4, lines 25-27 and page 20, line 24 to page 21, lines 2 of the specification, it is clear that "typical usage" relates to 'typical' stimuli to which the application is likely to be exposed, and not to the point in time relative to other program modules at which a code portion is used. The concept disclosed in the specification is entirely missing from Blake, and Blake does not even use the terminology "typical usage" anywhere, much less in relation to the terminology "concurrency of usage." Just because both of the terms "typical usage" and "concurrency of usage" include the word "usage" does not mean that they are equivalent, and in fact, as the terms "typical" and "concurrent" have different meanings, there must be something more than the terms themselves to support the contention that "typical usage" means the same thing as "concurrency of usage." It is clear from Blake that there is nothing more to support that contention.

In regards to claim 5, the Examiner again cites to a portion of Blake that describes a process performed during compilation, whereas the claimed element of "porting the annotated computer program" occurs prior to compilation of the source code on the target computer platform into an executable. Blake is simply not relevant to the claimed invention.

In regards to claim 6, the second plurality of directives indicates linkages between program objects in the source code, whereas the Examiner has again cited to a process that is performed during compilation. Compilation occurs after the second plurality of directives are

**ATTORNEY DOCKET NO.  
13644.0046**

**PATENT**

placed in the source code. A process performed on executable or object code is simply not applicable to a process performed on source code. Blake is simply not relevant to the claimed invention.

In regards to claim 7, not only does the Examiner again rely on a section of Blake that pertains to a process performed on object code as anticipating a claim element that is performed on source code, but the Examiner also relies on inherency without explaining why the element is necessarily present in the prior art. The Examiner asserts that the "compiler inherently identifies the boundaries between code portions, since it inserts calls to the library routine into each code portion," however this is not sufficient to establish that the compiler *necessarily* identifies boundaries. The cited section of the specification does not state that the compiler identifies any boundaries between code portions – it is clear that a "code portion" is defined by the presence of a call to the library routine, and that there is simply no need for a boundary to be identified by the compiler of Blake. In that regard, it is noted that the term "boundary" is not even used anywhere in Blake, much less in the cited section. Blake is simply not relevant to the claimed invention.

In regards to claim 8, the Examiner asserts that Blake discloses each of the program objects having a unique name, but then states that this teaching is inherent. There is a basic, fundamental difference between something that is disclosed and something that is inherent, both in accordance with the ordinary usage of those terms and as a matter of law. Blake fails to disclose that each of the program objects has a unique name in either regard – Blake does not state that (indeed, the term "name" is not even used in Blake), nor is that necessarily required by Blake. It is entirely possible that the program objects of Blake do *not* have a unique name, much less *any* name. A name is not required for the program objects of Blake in order for the process of Blake to operate. Blake is simply not relevant to the claimed invention. Withdrawal of the rejection of claims 1-3 and 5-8 under 35 U.S.C. 102 over Blake is respectfully requested.

**Rejections under 35 U.S.C. 103**

Claim 4 stands rejected under 35 U.S.C. 103(a) as being unpatentable over Blake in view of Mattson. Claims 9 and 10 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Blake in view of Komatsu. These rejections are respectfully traversed.

**ATTORNEY DOCKET NO.  
13644.0046**

**PATENT**

In addition to the numerous reasons identified above why Blake fails to anticipate the claimed invention, Mattson is non-analogous art from an entirely different class than Blake, and the combination of Blake and Mattson fails to provide a prima facie basis for the rejection of claim 4. In particular, it is incorrectly assumed that "static" and "overlay" are opposites. As used by those in the art and in the exemplary embodiments in the specification, "overlay" means "paged as needed," see page 5 of the specification at line 1. "Dynamic" has an ordinary meaning of "time varying," and is not used in the specification in regards to "overlay," if at all. A program that is present at all times and not paged as needed but which varies as a function of time would be "dynamic," and as such, would be different from an overlay program that can be static, i.e., not changing over time, and that can also be paged as needed. The construction of the term "overlay" as being the same as "dynamic" is simply improper, but the combination of Blake and Mattson would nevertheless fail to provide a basis for the rejection of claim 4 even if those terms were equivalent.

Likewise, in regards to claims 9 and 10, the combination fails to provide a prima facie basis for the rejection of the claims. In addition to the numerous reasons above why Blake fails to anticipate the claimed invention, the cited section of Komatsu is clear that the first computer compiles the code, which is then transferred to the client computer without any regard to the specific design of the second computer. In contrast, claim 9 states that the first computer "allocate[s] the program objects to fixed locations in the memory of the second computer" and "generate[s] an executable image . . . configured for execution on the second computer." Thus, in addition to the reasons provided above, the combination of Blake and Komatsu fails to provide a prima facie basis for the rejection of claims 9 and 10.

**ATTORNEY DOCKET NO.**  
**13644.0046**

**PATENT**

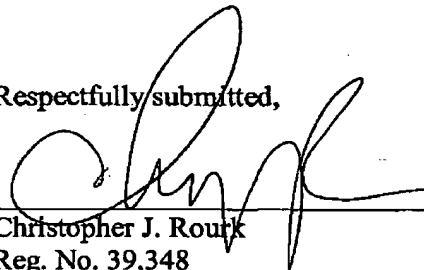
**CONCLUSION**

In view of the foregoing remarks and for various other reasons readily apparent, Applicant submits that all of the claims now present are allowable, and withdrawal of the rejection and a Notice of Allowance are courteously solicited.

If any impediment to the allowance of the claims remains after consideration of this amendment, a telephone interview with the Examiner is hereby requested by the undersigned at (214) 939-8657 so that such issues may be resolved as expeditiously as possible.

No fee is believed to be due at this time. If any applicable fee or refund has been overlooked, the Commissioner is hereby authorized to charge any fee or credit any refund to the deposit account of Godwin Gruber LLP, No. 500530.

Respectfully submitted,



Christopher J. Rourke  
Reg. No. 39,348  
ATTORNEY FOR APPLICANT

Date: February 28, 2005

GODWIN GRUBER LLP  
1201 Elm Street, Suite 1700  
Dallas, TX 75270  
Direct: 214-939-8657  
Fax: 214-760-7332  
Email: [crourke@godwingruber.com](mailto:crourke@godwingruber.com)